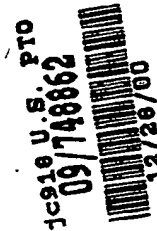


IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of:	)	
	)	
Satoshi SAKAMOTO, et al.	)	
	)	Group Art Unit: Unassigned
Serial No.: To be assigned	)	
	)	Examiner: Unassigned
Filed: December 28, 2000	)	



For: METHOD FOR CONTROLLING MULTITHREADING

**SUBMISSION OF CERTIFIED COPY OF PRIOR FOREIGN  
APPLICATION IN ACCORDANCE  
WITH THE REQUIREMENTS OF 37 C.F.R. §1.55**

*Assistant Commissioner for Patents  
Washington, D.C. 20231*

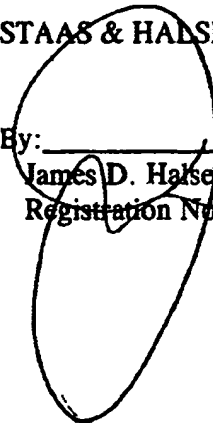
*Sir:*

In accordance with the provisions of 37 C.F.R. §1.55, the applicants submit herewith a certified copy of the following foreign application:

Japanese Patent Application No. 2000-016070  
Filed: January 25, 2000.

It is respectfully requested that the applicants be given the benefit of the foreign filing date as evidenced by the certified papers attached hereto, in accordance with the requirements of 35 U.S.C. §119.

Respectfully submitted,  
STAAS & HALSEY LLP

By:   
James D. Halsey, Jr.  
Registration No. 22,729

700 11th Street, N.W., Ste. 500  
Washington, D.C. 20001  
(202) 434-1500  
Date: December 28, 2000

【書類名】 特許願

【整理番号】 9951413

【提出日】 平成12年 1月25日

【あて先】 特許庁長官殿

【国際特許分類】 G06F 9/46

【発明の名称】 マルチスレッド制御方法、マルチスレッド制御装置、及び記録媒体

【請求項の数】 6

【発明者】

    【住所又は居所】 神奈川県川崎市中原区上小田中4丁目1番1号 富士通株式会社内

    【氏名】 坂本 智

【発明者】

    【住所又は居所】 神奈川県川崎市中原区上小田中4丁目1番1号 富士通株式会社内

    【氏名】 岩崎 正則

【発明者】

    【住所又は居所】 神奈川県川崎市中原区上小田中4丁目1番1号 富士通株式会社内

    【氏名】 坪井 晃

【特許出願人】

    【識別番号】 000005223

    【氏名又は名称】 富士通株式会社

【代理人】

    【識別番号】 100068755

    【住所又は居所】 岐阜市大宮町2丁目12番地の1

    【弁理士】

    【氏名又は名称】 恩田 博宣

    【電話番号】 058-265-1810

【選任した代理人】

【識別番号】 100105957

【住所又は居所】 東京都渋谷区代々木二丁目 1 0 番 4 号 新宿辻ビル 8  
階

【弁理士】

【氏名又は名称】 恩田 誠

【電話番号】 03-5365-3057

【手数料の表示】

【予納台帳番号】 002956

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 9909792

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 マルチスレッド制御方法、マルチスレッド制御装置、及び記録媒体

【特許請求の範囲】

【請求項 1】 複数のスレッドによって並列処理を行うマルチスレッド制御方法において、

並列処理を行っている実行中スレッドの数と待機状態にある待機スレッドの数を監視し、両スレッド数に基づいて不要な数の待機スレッドを回収することを特徴とするマルチスレッド制御方法。

【請求項 2】 所定時間毎に、前記待機スレッドの数と、前記所定時間内の実行中スレッドの数の最大値、平均値、及びそれらに所定の係数を乗じた値のうちの何れか一つからなる必要数とを比較し、前記待機スレッドの数が前記必要数より多い場合に該必要数まで前記待機スレッドを回収することを特徴とする請求項 1 記載のマルチスレッド制御方法。

【請求項 3】 複数のスレッドによって並列処理を行うマルチスレッド制御装置において、

前記複数のスレッドのスレッド情報が記憶されるスレッド情報管理部と、

前記スレッド情報管理部に記憶された待機状態にあるスレッドの数に基づいてスレッドの生成要求、待機状態にあるスレッドに対して処理の実行要求を行うスレッド実行処理部と、

前記スレッド情報管理部に記録された並列処理を行っている実行中スレッドの数と待機状態にある待機スレッドの数に基づいて不要な数の待機スレッドを回収するスレッド回収処理部と

を備えたことを特徴とするスレッド制御装置。

【請求項 4】 前記スレッド回収処理部は、

所定時間毎に、前記待機スレッドの数と、前記所定時間内の実行中スレッドの数の最大値、平均値、及びそれらに所定の係数を乗じた値のうちの何れか一つからなる必要数とを比較する手段と、

前記待機スレッドの数が前記必要数より多い場合に該必要数まで前記待機スレ

ッドを回収する手段と

を備えたことを特徴とする請求項 3 記載のスレッド制御装置。

【請求項 5】 複数のスレッドによって並列処理を行うマルチスレッド制御プログラムが記録されたコンピュータ読み取り可能な記録媒体であって、

前記プログラムは、

並列処理を行っている実行中スレッドの数と待機状態にある待機スレッドの数を監視するステップと、

両スレッド数に基づいて不要な数の待機スレッドを回収するステップ  
を実行する、記録媒体。

【請求項 6】 前記プログラムは、

所定時間毎に、前記待機スレッドの数と、前記所定時間内の実行中スレッドの数の最大値、平均値、及びそれらに所定の係数を乗じた値のうちの何れか一つからなる必要数とを比較するステップと、

前記待機スレッドの数が前記必要数より多い場合に該必要数まで前記待機スレッドを回収するステップ  
を実行する、請求項 5 記載の記録媒体。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、複数のスレッドによって並列処理を行うシステムにおいて、プログラムの実行速度の向上とシステム資源の有効活用を図ることができるマルチスレッド制御方法、その装置、及び記録媒体に関するものである。

【0002】

近年、科学技術計算等のプログラムを高速に実行するシステムのオペレーティングシステムプログラムには並列動作機構が備えられているものがある。このようなオペレーティングシステムは、プログラムを所定の処理で分割し、各処理をシステム資源を割り当てたスレッドにて平行実行することで、プログラムの実行速度を高くしている。このようなシステムにおいて、プログラム実行速度の高速化とシステム資源の有効活用が求められている。

## 【 0 0 0 3 】

## 【従来の技術】

従来、科学技術計算等のプログラムを実行するシステムでは、プログラムの並列実行などによってプログラムの実行速度の高速化を図る等を目的にスレッドが用いられる。システムのオペレーティングシステム（OS）プログラムは、アプリケーションに応じてスレッドを生成する。生成した各スレッドにシステム資源を割り当てることで、各スレッドに対応させたプログラムを並列実行し、プログラムの実行速度を高くしている。

## 【 0 0 0 4 】

しかし、OSプログラムのスレッド生成には時間がかかるため、一度生成したスレッドを削除することなく再利用することで、システムにおける処理の高速化を図ることが行われている。

## 【 0 0 0 5 】

詳述すると、プログラムの実行を終了したスレッドを終了させずに待機スレッドとしてテーブルに登録する。そして、スレッドの生成要求があった場合には、テーブルに登録した待機スレッドを用いてプログラムを実行する。これにより、スレッド生成時間を短縮・削減する。

## 【 0 0 0 6 】

## 【発明が解決しようとする課題】

従来のマルチスレッド制御方法では、生成されたスレッドの数だけ待機スレッドとして管理されており、実際に動作しているスレッドが少ない状態でも、不要な待機スレッドがシステムに沢山存在している。

## 【 0 0 0 7 】

待機スレッドにもシステム資源が割り当てられている。即ち、待機スレッドは、その待機中でもメモリを使用している。このように、待機スレッドが無駄にメモリ資源を使用している状況に陥ることがある。

## 【 0 0 0 8 】

例えば、ある時刻において200個のスレッドが並列に動作し、その後2個のスレッドしか動作しない場合、198個の待機スレッドがメモリを使用すること

になる。システム資源の少ないシステムにおいて、このような状況はシステムの運営上好ましくなく、他のプログラムの実行速度にも大きな影響を与える。

#### 【0009】

本発明は上記問題点を解決するためになされたものであって、その目的はプログラム実行速度の高速化とシステム資源の有効活用を図ることのできるマルチスレッド制御方法その装置、及び記録媒体を提供することにある。

#### 【0010】

##### 【課題を解決するための手段】

上記目的を達成するため、本発明は、並列処理を行っている実行中スレッドの数と待機状態にある待機スレッドの数を監視し、両スレッド数に基づいて不要数の待機スレッドを回収する。これにより、無駄にメモリ資源を使用している待機スレッドを終了させ、システム資源の有効活用が図られる。

#### 【0011】

また、所定時間毎に、前記待機スレッドの数と、前記所定時間内の実行中スレッドの数の最大値、平均値、及びそれらに所定の係数を乗じた値のうちの何れか一つからなる必要数とを比較し、前記待機スレッドの数が前記必要数より多い場合に該必要数まで前記待機スレッドを回収する。これにより、稼働状況に応じた必要数にまで待機スレッドの数を低減してシステム資源の有効活用が図られる。

#### 【0012】

##### 【発明の実施の形態】

以下、本発明を具体化した一実施の形態を図1～図6に従って説明する。

図1は、一実施形態のマルチスレッド制御装置の概略構成図である。

#### 【0013】

マルチスレッド制御装置1は、アプリケーションプログラムを実行するシステム（計算機）に備えられ、オペレーティングシステム（OS）2にスレッド生成を要求し、OS2から受け取ったスレッドを用いてプログラムの並列可能な部分プログラム（例えばDOLープやサブルーチン）を並列実行する。

#### 【0014】

マルチスレッド制御装置1は、部分プログラムの終了後、不要となったスレ

ドを待機スレッドとして登録する。次に、プログラムの実行要求があると、マルチスレッド制御装置 1 は、待機スレッドの状態を見て、待機スレッドがあればそれを用いて部分プログラムを実行し、待機スレッドが無い場合には OS 2 にスレッド生成を要求する。

#### 【 0 0 1 5 】

マルチスレッド制御装置 1 は、プログラム実行中のスレッドの数と待機スレッドの数を監視し、実行中スレッド数に基づいて、必要数以上の待機スレッドを回収する。例えば、マルチスレッド制御装置 1 は、最大実行中スレッド数と待機スレッド数を用いて、待機スレッドの回収処理を実行する。

#### 【 0 0 1 6 】

詳述すると、マルチスレッド制御装置 1 は、実行中のスレッドの数を所定期間監視し、その期間内の最大値（最大実行中スレッド数）をその時に於ける待機スレッドの必要数とし、それと現在管理している待機スレッドの数とを比較する。そして、マルチスレッド制御装置 1 は、最大実行中スレッド数よりも待機スレッド数が大きい場合、最大実行中スレッド数まで待機スレッドを回収する。即ち、マルチスレッド制御装置 1 は、待機スレッド数が最大実行中スレッド数と一致するまで待機スレッドを終了させる。

#### 【 0 0 1 7 】

このように、無駄にシステム資源を使用している待機スレッドを回収することで、実行中のスレッドに与える影響を少なくする。

また、最大実行中スレッド数は、実行しているプログラムの状態、即ち、システムの稼働状況に応じて変化する。従って、最大実行中スレッド数まで待機スレッドを回収することで、システムの稼働状況に応じて待機スレッド数を動的に変化させている。

#### 【 0 0 1 8 】

更に、所定期間実行中スレッドを監視した最大実行中スレッド数にまで待機スレッドを回収することで、不意なスレッドの増加に対応することができる。即ち、実行中にスレッド数が一時的に低下しても、最大実行中スレッド数だけ待機スレッドを待機させているため、待機スレッドが不足することが無く、プログラム



を直ぐに実行することができる。

【0019】

一方、待機スレッド数が実行中スレッド数と等しくなるように待機スレッドを回収する方法では、実行中スレッド数の一時的な低下に対応して待機スレッド数を低減させるため、次にプログラムに応じてスレッドが必要となった場合に待機スレッドが不足してスレッドの生成要求をOS2に行わなければならない、その分実行時間が長くなる（実行開始が遅れる）。

【0020】

次に、マルチスレッド制御装置1の構成を説明する。

マルチスレッド制御装置1は、スレッド管理処理部11、スレッド回収処理部12、スレッド管理テーブル13を含む。

【0021】

スレッド管理処理部11は、スレッド生成をOSに要求する処理、及びスレッドにプログラムの実行を要求する処理を実行する。スレッド回収処理部12は、待機スレッドを監視し、無駄な待機スレッドを回収（スレッド終了）する処理を実行する。スレッド管理テーブル13は、スレッド情報を管理するためのテーブルである。

【0022】

尚、図1には、OS2にて生成されたスレッドとして3つのスレッド14, 15, 16を示す。これらスレッド14～16のうち、スレッド14, 15はプログラムを実行している実行中スレッドであり、スレッド16はプログラムを終了して待機している待機スレッドとして両処理部11, 12の動作及びテーブル13の構成を説明する。

【0023】

先ず、スレッド管理テーブル13の構成を図3, 4に従って説明する。

図4は、スレッド管理テーブル13の概略構成図である。

スレッド管理テーブル13は、情報管理テーブル21と複数（図4では、図1のスレッド14～16に対応して3つ）のスレッド制御表22～24を含む。情報管理テーブル21は、実行中スレッドキュー21a、待機スレッドキュー21

b、待機スレッドカウンタ 2 1 c、実行中スレッドカウンタ 2 1 d、最大実行中スレッドカウンタ 2 1 e のそれぞれのための領域を持つ。

【 0 0 2 4 】

実行中スレッドキュー 2 1 a は、それにキューイングされた実行中スレッドに対応するスレッド制御表を指すアドレスを記憶する領域を持つスレッド制御表アドレスリストであり、実行中のスレッドの情報を管理する。待機スレッドキュー 2 1 b は、それにキューイングされた待機スレッドに対応するスレッド制御表を指すアドレスを記憶する領域を持ち、待機状態のスレッドの情報を管理する。詳述すると、図 1 においてスレッド 1 4、1 5 が実行中スレッド、スレッド 1 6 が待機スレッドである。これに対し、実行中スレッドキュー 2 1 a には実行中スレッド 1 4、1 5 に対応するスレッド制御表 2 2、2 3 を指すアドレスが記録され、待機スレッドキュー 2 1 b には待機スレッド 1 6 に対応するスレッド制御表 2 4 を指すアドレスが記録されている。

【 0 0 2 5 】

図 3 は、スレッド制御表 2 4 の構成を示す説明図である。尚、他のスレッド制御表 2 2、2 3 の構成はスレッド制御表 2 4 のそれと同じであるため、図面及び説明を省略する。

【 0 0 2 6 】

スレッド制御表 2 4 は、スレッド実行 E C B (Event Control Block) 2 4 a、スレッド停止 E C B 2 4 b のための領域を持つ。スレッド実行 E C B 2 4 a は図 1 のスレッド管理処理部 1 1 からの実行要求を管理するために設けられ、スレッド停止 E C B 2 4 b は図 1 のスレッド回収処理部 1 2 からの終了要求を管理するために設けられている。図 1 の待機スレッド 1 6 は、各 E C B 2 4 a、2 4 b に書き込まれた情報に基づいてプログラムの実行又は停止処理を行う。

【 0 0 2 7 】

詳述すると、図 1 のスレッド管理処理部 1 1 は、待機スレッドキュー 2 1 b にキューイングされたスレッド制御表 2 4 のスレッド実行 E C B 2 4 a に実行要求に応じた情報を格納する。待機スレッド 1 6 は、自己に対応したスレッド制御表 2 4 のスレッド実行 E C B 2 4 a に実行要求に応じた情報が格納されると、それ

に応答してスレッドに割り当てられたプログラム部分を実行する。

【 0 0 2 8 】

図 1 のスレッド回収処理部 1 2 は、待機スレッドキュー 2 1 b にキューイングされたスレッド制御表 2 4 のスレッド停止 E C B 2 4 b に停止要求に応じた情報を格納する。待機スレッド 1 6 は、スレッド停止 E C B 2 4 b に停止要求に応じた情報が格納されると、それに応答して終了処理を実行する。これにより、待機スレッド 1 6 に割り当てられていたシステム資源が解放される。

【 0 0 2 9 】

待機スレッドカウンタ 2 1 c には、待機状態のスレッド数が記録され、実行中スレッドカウンタ 2 1 d には、実行中のスレッド数が記録され、最大実行中スレッドカウンタ 2 1 e には実行中のスレッド数の最大値が記録される。

【 0 0 3 0 】

図 1 の実行中スレッド 1 4, 1 5 は、プログラムを終了すると、実行中スレッドカウンタ 2 1 d をカウントダウン ( - 1 ) し、待機スレッドカウンタ 2 1 c をカウントアップ ( + 1 ) する。図 1 の待機スレッド 1 6 は、スレッド管理処理部 1 1 から実行要求を受け取ると、待機スレッドカウンタ 2 1 c をカウントダウンし、実行中スレッドカウンタ 2 1 d をカウントアップし、そのカウンタ 2 1 d の値が最大実行中スレッドカウンタ 2 1 e の値よりも大きい場合、最大実行中スレッドカウンタ 2 1 e の値を実行中スレッドカウンタ 2 1 d の値に更新する。このように、両カウンタ 2 1 c, 2 1 d は、待機中のスレッド数と実行中のスレッド数を表す。

【 0 0 3 1 】

従って、図 1 のスレッド管理処理部 1 1 は、待機スレッドカウンタ 2 1 c に基づいて、スレッド生成要求、スレッドに対する実行要求を行う。詳述すると、スレッド管理処理部 1 1 は、並列のプログラムの実行要求を受けると、待機スレッドカウンタ 2 1 c が「 0 」の場合には O S 2 にスレッド生成を要求し、生成されたスレッドを実行中スレッドキュー 2 1 a にキューイングし、そのスレッドにプログラムの実行を要求する。一方、カウンタ 2 1 c が「 1 」以上の場合、スレッド管理処理部 1 1 は、待機スレッドキュー 2 1 b にキューイングされた待機スレ

ッド 1 6 にプログラムの実行を要求する。

【 0 0 3 2 】

また、図 1 のスレッド回収処理部 1 2 は、待機スレッドカウンタ 2 1 c と最大実行中スレッドカウンタ 2 1 e の値に基づいて、余分なスレッドを回収する。詳述すると、スレッド回収処理部 1 2 は、最大実行中スレッドカウンタ 2 1 e と待機スレッドカウンタ 2 1 c を所定の時間毎にモニタし、それらを比較する。そして、スレッド回収処理部 1 2 は、比較結果に基づいて、最大実行中スレッドカウンタ 2 1 e の値より待機スレッドカウンタ 2 1 c のそれの方が大きい場合に、最大実行中スレッドカウンタ 2 1 e の値まで待機スレッドを回収する。即ち、スレッド回収処理部 1 2 は、両カウンタ 2 1 c, 2 1 e の値の差の数のスレッドに対して終了要求を行い、待機スレッドを回収する。

【 0 0 3 3 】

次に、スレッド 1 4 ~ 1 6 及びスレッド回収処理部 1 2 の動作を図 5, 図 6 のフローチャートに従って説明する。

図 5 は、スレッドが実行する処理を説明する動作フローチャートである。

【 0 0 3 4 】

まず、ステップ 3 1 において、スレッドは、実行中スレッドカウンタ 2 1 d を + 1 する。

次に、ステップ 3 2 において、スレッドは実行中スレッドカウンタ 2 1 d と最大実行中スレッドカウンタ 2 1 e の値を比較し、実行中スレッドカウンタ 2 1 d の方が最大実行中スレッドカウンタ 2 1 e より大きい場合はステップ 3 3 へ移る。そのステップ 3 3 において、スレッドは最大実行中スレッドカウンタ 2 1 e の値を実行中スレッドカウンタ 2 1 d のそれに更新する。

【 0 0 3 5 】

ステップ 3 4 において、スレッドはプログラムを実行する。そのプログラムが終了すると、スレッドはステップ 3 5 に移る。

ステップ 3 5 において、スレッドは実行中スレッドカウンタ 2 1 d を - 1 する。次に、ステップ 3 6 において、スレッドは待機スレッドカウンタ 2 1 c を + 1 する。そして、ステップ 3 7 において、スレッドは待機スレッドキュー 2 1 b に

スレッド制御表をキューイングする。

【 0 0 3 6 】

ステップ 3 8 において、スレッドはマルチ W A I T 処理を実行する。この処理において、スレッドは、スレッド実行 E C B 2 4 a とスレッド停止 E C B 2 4 b の両方を監視し、何れか一方に情報が書き込まれるまで待つ。そして、スレッドは、スレッド実行 E C B 2 4 a またはスレッド停止 E C B 2 4 b に情報が書き込まれると、ステップ 3 9 に移る。

【 0 0 3 7 】

ステップ 3 9 において、スレッドは、ステップ 3 8 において書き込まれた情報が終了要求か否かを判断し、終了要求ではない、即ち実行要求の場合にはステップ 4 0 に移る。そのステップ 4 0 において、スレッドは、待機スレッドカウンタ 2 1 c を - 1 した後、ステップ 3 1 に移る。これにより、スレッドはプログラムを実行するスレッドとして機能し、待機スレッドが再利用されることになる。

【 0 0 3 8 】

一方、ステップ 3 9 において、入力終了要求の場合、スレッドはステップ 4 1 に移り、そのステップ 4 1 において終了処理を実行する。これより、スレッドが終了（消滅）する。

【 0 0 3 9 】

図 6 は、スレッド回収処理部 1 2 の動作フローチャートである。

先ず、ステップ 5 1 において、スレッド回収処理部 1 2 は、所定時間のタイマをかける。この所定時間は、待機スレッドカウンタ 2 1 c と最大実行中スレッドカウンタ 2 1 e をモニタする間隔であり、このモニタ間隔はシステムに応じて設定した一定値、又は稼働時刻等に対応して変更してもよい。そして、所定時間経過してタイムアップすると、スレッド回収処理部 1 2 はステップ 5 2 に移る。

【 0 0 4 0 】

ステップ 5 2 は比較処理（比較手段）であり、スレッド回収処理部 1 2 は、待機スレッドカウンタ 2 1 c と最大実行中スレッドカウンタ 2 1 e とを比較し、待機スレッド数が最大実行中スレッド数より大きくなっていないか確認する。そして、スレッド回収処理部 1 2 は、待機スレッド数より最大実行中スレッド数の方

が大きい場合にはステップ 5 1 に戻り、待機スレッド数の方が最大実行中スレッド数より大きい場合にはステップ 5 3 に進む。

【 0 0 4 1 】

そのステップ 5 3 は回収処理（回収手段）であり、スレッド回収処理部 1 2 は、待機スレッドに終了要求を行い（詳しくは待機スレッドに対応するスレッド制御表のスレッド E C B 2 4 b に終了要求に応じた情報を書き込む）、スレッドを回収する。

【 0 0 4 2 】

上記マルチスレッド制御装置 1 は、計算機上で実行するコンピュータプログラムにより実現することもできる。

即ち、図 2 に示すように、マルチスレッドにてプログラムを並列実行する計算機 6 0 は、処理装置 6 1、入出力装置 6 2、主記憶装置 6 3、補助記憶装置 6 4 などから構成され、コンピュータプログラムを実行するものである。コンピュータプログラムは、フロッピーディスクや C D - R O M 等の可搬型記録媒体 6 5 やネットワーク接続された他の計算機の主記憶装置や補助記憶装置等に格納されて提供される。

【 0 0 4 3 】

提供されたコンピュータプログラムは、可搬型媒体 6 5 から一旦補助記憶装置 6 4 にコピーまたはインストール後に主記憶装置 6 3 にロードされ、または可搬型媒体 6 5 から直接計算機 6 0 の主記憶装置 6 3 にロードされ実行する。尚、主記憶装置 6 3 には、図 1 のスレッド管理テーブル 1 3 が作成され、それによりスレッド情報が管理される。

【 0 0 4 4 】

また、コンピュータプログラムがネットワーク接続された他の装置に格納されて提供された場合も、他の装置からネットワークを経由して受信後に補助記憶装置 6 4 にコピーまたはインストール後に主記憶装置 6 3 にロードされ、または直接主記憶装置 6 3 にロードされ実行するものである。

【 0 0 4 5 】

尚、図 2 は、コンピュータプログラムを実行する計算機を機能的に示したもの

であり、各装置 6 1 ～ 6 4 が複数設けられた計算機を用いる、又は複数の計算機が疎又は密結合された計算機システムを用いても良い。

【 0 0 4 6 】

以上記述したように、本実施の形態によれば、以下の効果を奏する。

(1) マルチスレッド制御装置 1 のスレッド回収処理部 1 2 は、実行中のスレッドの数を所定期間監視し、その期間内の最大値（最大実行中スレッド数）と待機スレッド数とを比較する。そして、スレッド回収処理部 1 2 は、最大実行中スレッド数よりも待機スレッド数が多い場合、最大実行中スレッド数まで待機スレッドを回収するようにした。その結果、無駄にシステム資源を使用している待機スレッドを回収し、実行中のスレッドに与える影響を少なくすることができる。

【 0 0 4 7 】

(2) スレッド回収処理部 1 2 は、最大実行中スレッドカウンタ 2 1 e の値、即ち所定期間に於ける実行中スレッド数の最大値まで待機スレッドを回収する。この最大実行中スレッド数は、実行しているプログラムの状態、即ち、システムの稼働状況に応じて変化する。その結果、最大実行中スレッド数まで待機スレッドを回収することで、システムの稼働状況に応じた数の待機スレッドを待機させることができる。

【 0 0 4 8 】

尚、前記実施形態は、以下の態様に変更してもよい。

○上記実施形態では、所定時間毎に待機スレッドカウンタ 2 1 c と最大実行中スレッドカウンタ 2 1 e を比較した結果に基づいて待機スレッドを終了させるようにしたが、待機スレッドカウンタ 2 1 c の値をその時における待機スレッドの必要数とし、それと実行中スレッドカウンタ 2 1 d を比較した結果に基づいて待機スレッドを回収するようにしても良い。また、必要数として所定期間に於ける実行中スレッド数の平均値を求め、その平均スレッド数と待機スレッド数とを比較し、その比較結果に基づいて平均値又は現在実行中のスレッド数まで待機スレッドを回収するようにしてもよい。また、実行中スレッド数にシステムの状況に応じた値又は一定値の係数を演算した数を必要数とし、それまで待機スレッドを

回収するようにしてもよい。

【 0 0 4 9 】

○上記実施形態では、最大実行中スレッドカウンタ 2 1 e の更新を各スレッド 1 4 ～ 1 6 が行うようにしたが、その更新処理をスレッド管理処理部 1 1 又はスレッド回収処理部 1 2 が行うようにしてもよい。

【 0 0 5 0 】

【発明の効果】

以上詳述したように、本発明によれば、実行中のスレッドの数と待機状態にあるスレッドの数に基づいて不要な数の待機スレッドを回収するようにした。これにより、待機スレッドにてプログラム実行速度の高速化を図り、無駄にメモリ資源を使用している待機スレッドを終了させ、システム資源の有効活用を図ることができる。

【 0 0 5 1 】

また、所定時間毎に、必要数まで前記待機スレッドを回収するようにしたため、稼働状況に応じた必要数にまで待機スレッドの数を低減してシステム資源の有効活用を図ることができる。

【図面の簡単な説明】

【図 1】 一実施形態のマルチスレッド制御装置の概略構成図である。

【図 2】 システムのハードウェア構成を示す概略図である。

【図 3】 スレッド制御表の説明図である。

【図 4】 スレッド管理テーブルの説明図である。

【図 5】 スレッドの処理を示すフローチャートである。

【図 6】 スレッド回収処理部の処理を示すフローチャートである。

【符号の説明】

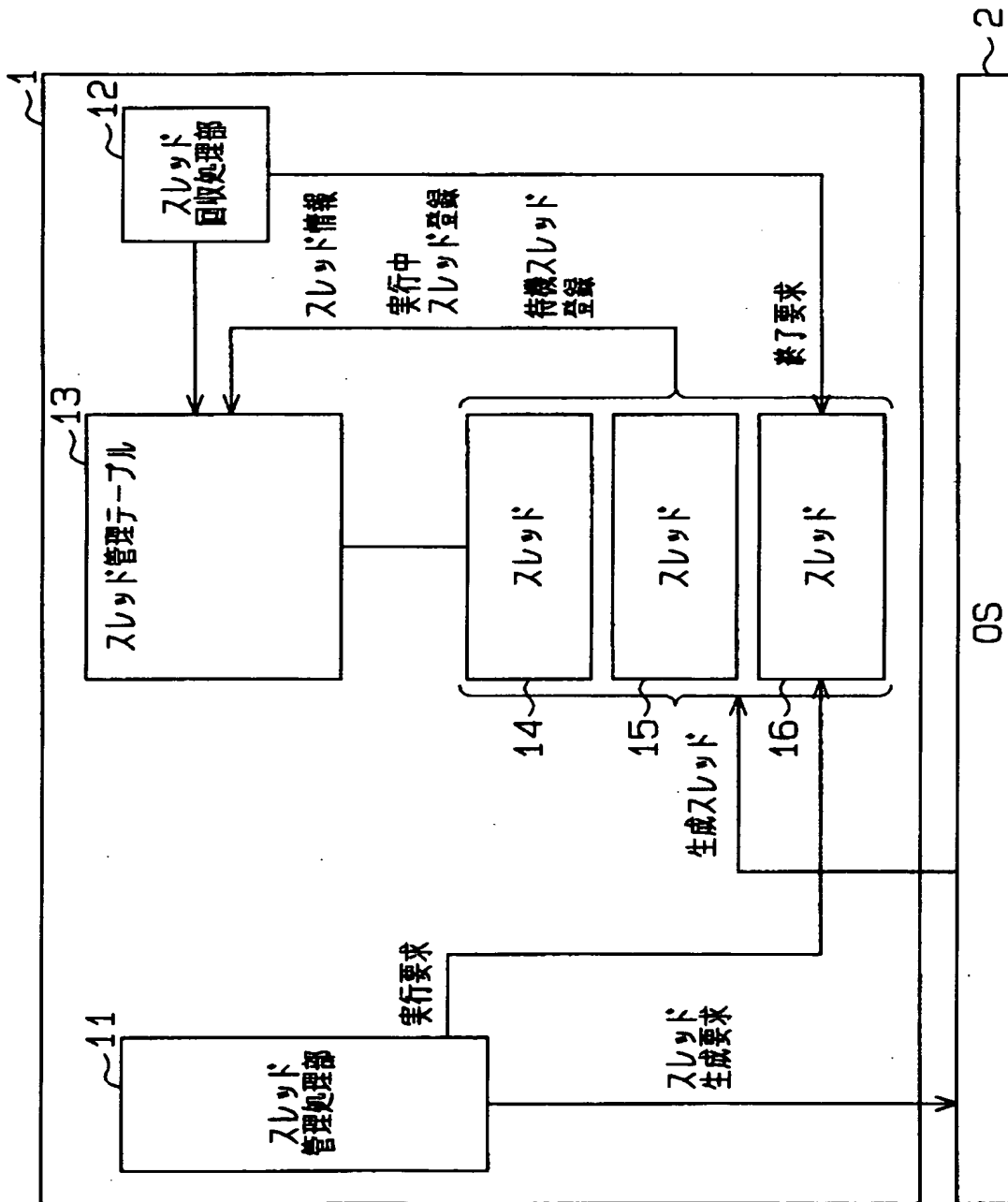
- 1 1 スレッド管理処理部
- 1 2 スレッド回収処理部
- 1 3 スレッド情報管理部としてのスレッド管理テーブル
- 1 4 ～ 1 6 スレッド



【書類名】 図面

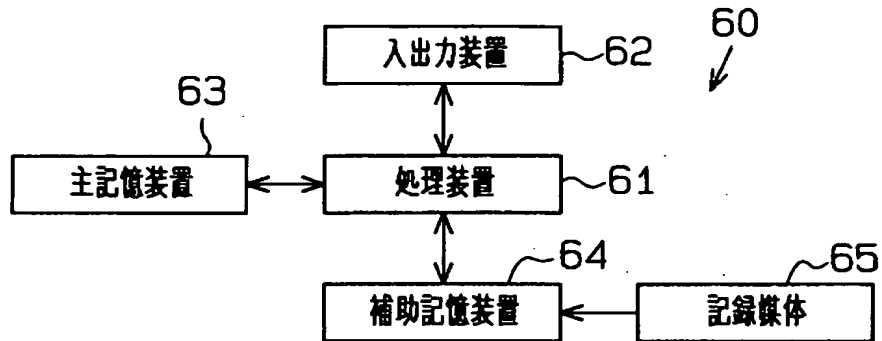
【図 1】

マルチスレッド制御装置の概略構成図



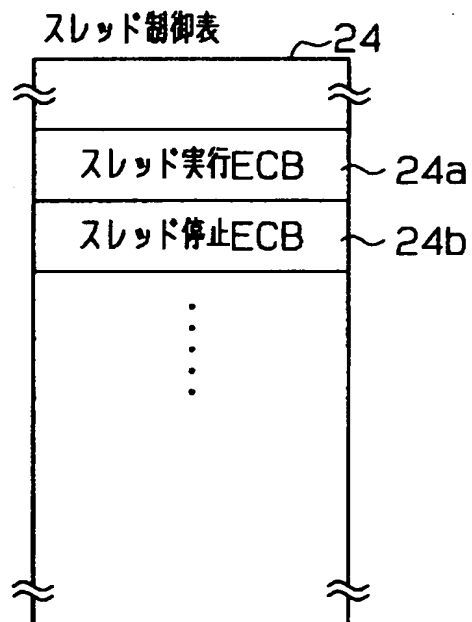
【図 2】

システムのハードウェア構成を示す概略図



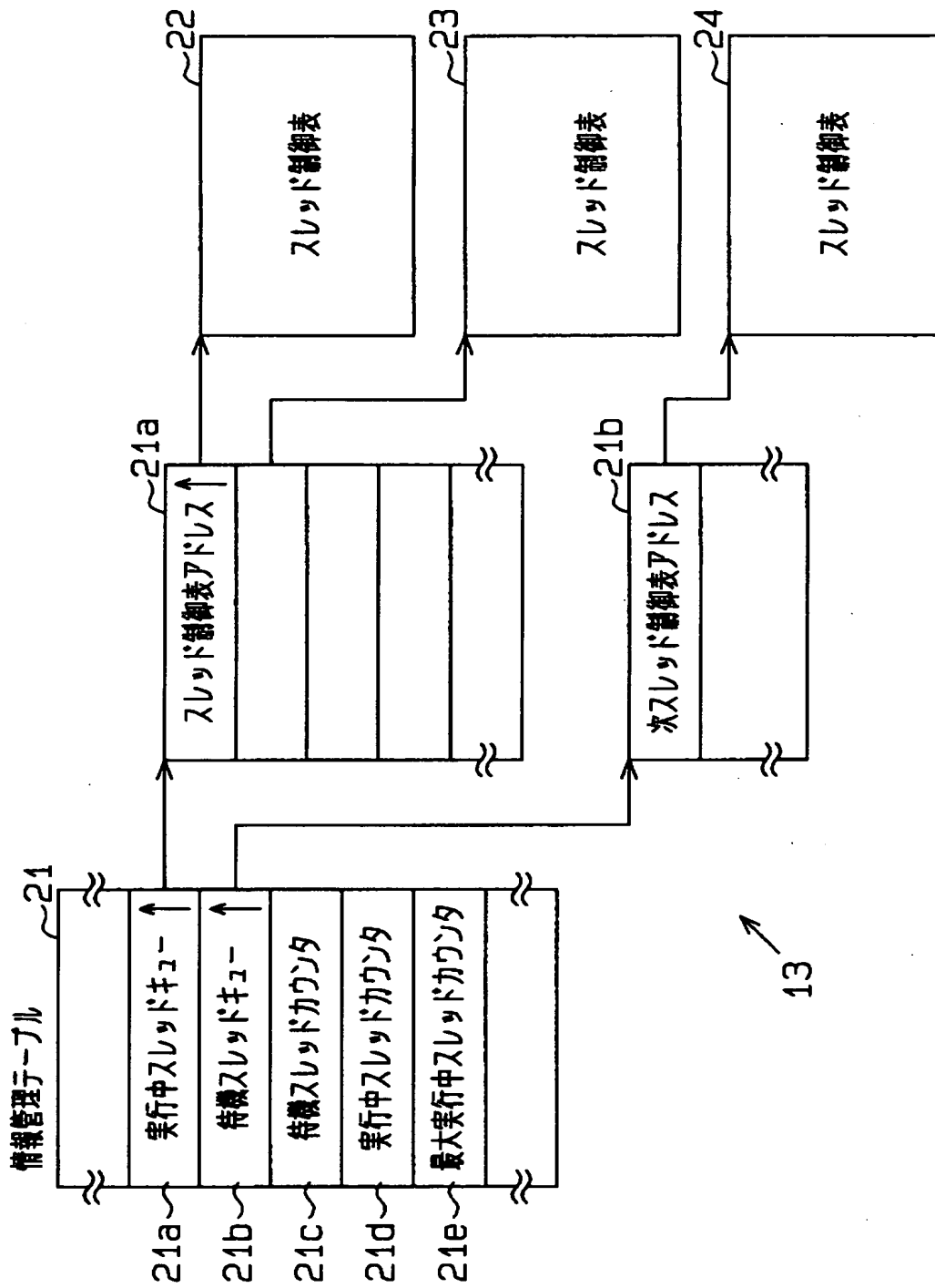
【図 3】

スレッド制御表の説明図



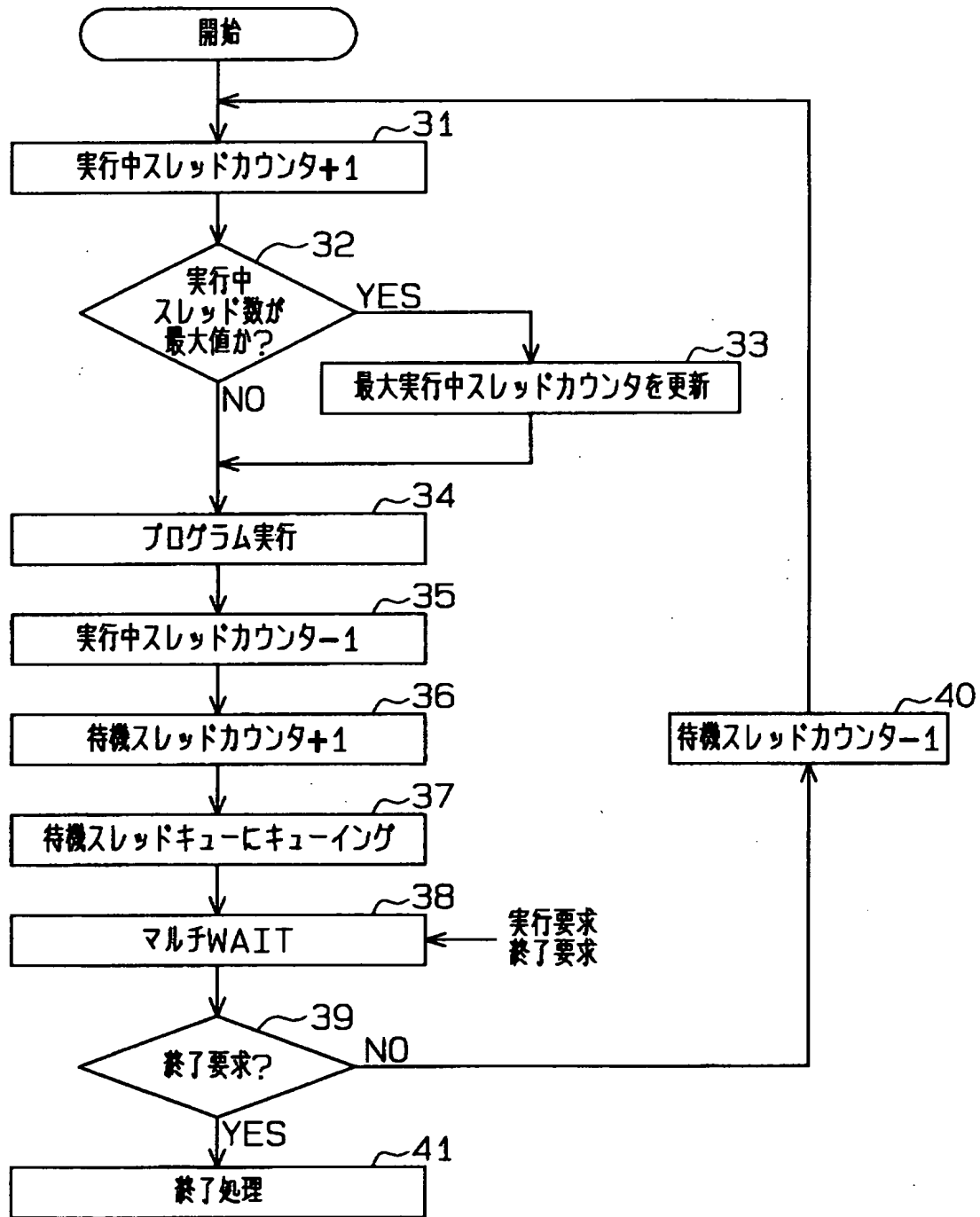
【図 4】

スレッド管理テーブルの説明図



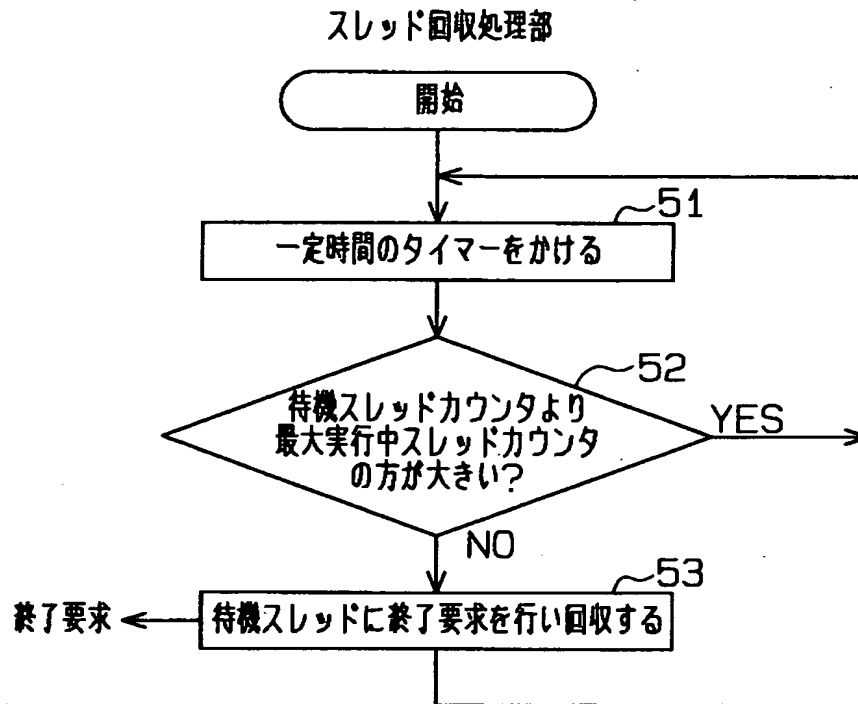
【図 5】

スレッドの処理を示すフローチャート



【図 6】

スレッド回収処理部の処理を示すフローチャート



【書類名】 要約書

【要約】

【課題】 プログラム実行速度の高速化とシステム資源の有効活用を図ることのできるマルチスレッド制御装置を提供すること。

【解決手段】 マルチスレッド制御装置 1 のスレッド回収処理部 1 2 は、実行中のスレッドの数を所定期間監視し、その期間内の最大値（最大実行中スレッド数）と待機スレッド数とを比較する。そして、スレッド回収処理部 1 2 は、最大実行中スレッド数よりも待機スレッド数が多い場合、最大実行中スレッド数まで待機スレッドを回収する。

【選択図】 図 1

出 願 人 履 歴 情 報

識別番号 [000005223]

1. 変更年月日	1996年 3月26日
[変更理由]	住所変更
住 所	神奈川県川崎市中原区上小田中4丁目1番1号
氏 名	富士通株式会社